Emre Tokyüz

BurgerApp Überblick

Die Applikation “BurgerApp” wurde entwickelt um die Programmiersprache “Swift” zu lernen und in Kombination mit SwiftUI, einem UI-Framework von Apple, eine Mobile Applikation zu entwickeln.
Der Logikteil der Applikation wurde vollständig gemäss den Use Cases (Siehe Burgerapp.pdf) implementiert. Alle Use Cases konnten im GUI durch Zeitdruck sowie Herausforderungen mit SwiftUI nicht vollständig implementiert werden.
Ein Video welches die App vorstellt kann hier gefunden werden.

Test Methode

Als Testing-Tool wurde Versucht die eingebaute Testing Suite von Xcode zu benutzen, da es sehr funktionsreich gewirkt hatte. Es wurde versucht, das Tutorial von Hacking with Swift zu implementieren, jedoch ging dies nicht problemlos. Die Klassen konnten erstellt werden, jedoch scheitert das aufnehmen der Klicks und der Simulator stürzt ab.
Deswegen wurden manuelle Tests durchgeführt. Zum einen wurde getestet, ob die vorhandene und getestete Programmlogik auch korrekt vom User Interface (UI) aufgerufen wird. Die User Experience (UX) wurde auch betrachtet und getestet.

User Interface

Der Programmablauf sollte wie in den Mockups verlaufen: Zuerst wird ein Startbildschirm gezeigt, bei dem der User sein Land auswählen kann (wichtig für die Preisberechnung). Daraufhin muss er auswählen, ob der Patty aus Fleisch ist oder Vegetarisch. Unten hat es einen Continue-Button, der grau ist wenn noch nichts ausgewählt wurde. Der Cart-Button zeigt die Anzahl an Burger in der jetzigen Bestellung auf und führt einem zum Checkout. Im nächsten Bildschirm kann der User seine Zutaten von einer Checkliste auswählen. Hier ändert der Continue-Button nie seinen Zustand, da jemand den Burger nur mit Buns und Patty bestellen darf. Der Cart-Button ist wieder vorhanden. Im nächsten und letztem Bildschirm werden dem Benutzer alle Burger angezeigt, deren Zutaten sowie den Preis pro Einheit. Unten steht das Total in der lokalen Währung. Es sollte Buttons haben um noch mehr Burger bestellen zu können. Da dies vom gleichen Land geschehen soll, wird man hier wieder zur “Select the Patty” Ansicht geführt und nicht an den Startbildschirm. Wenn man “Pay” drückt, gelangt man wieder zum Startbildschirm und darf von einem anderen Land aus bestellen. Idealerweise kann der Benutzer auch zurückgehen zwischen diesen Menüs, z.B falls er den falschen Patty-Typ ausgewählt hat. Er sollte aber nach der Wahl vom Land nicht wieder zurückgehen können.

Abbildung 1 Mockup

Test 1 Programmablauf funktioniert

Hier wird der zuvor beschriebene Programmablauf kontrolliert und getestet. Kriterium zum bestehen: Programmablauf ist: Startbildschirm -> Patty-Select -> Zutaten -> Checkout.

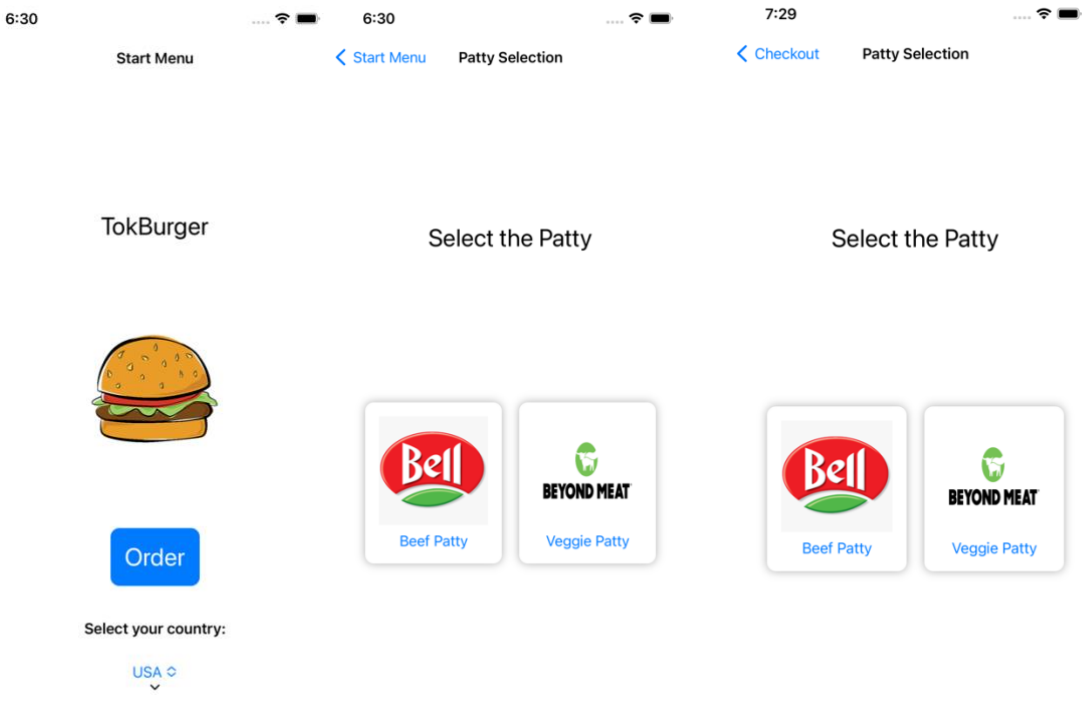


Abbildung 2 Startbildschirm wird angezeigt.

Abbildung 3 Select Patty bevor "Beef" ausgewählt wurde.

Abbildung 4 Select Patty nachdem "Beef" ausgewählt wurde.

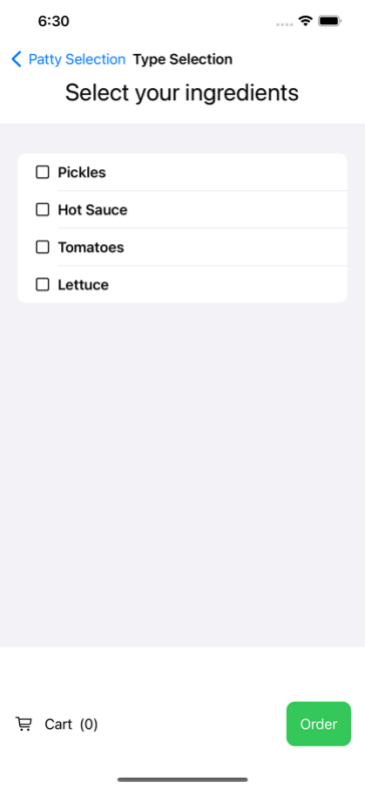


Abbildung 5 Zutatenbildschirm ohne etwas anzuklicken. (Pickles wurde nachher angeklickt)

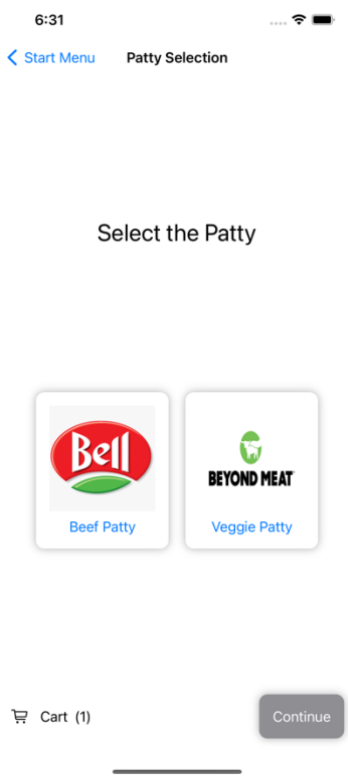


Abbildung 6 Nachdem man bei den Zutaten auf Order drückt, wird kurz das Checkout angezeigt und man landet daraufhin wieder hier.

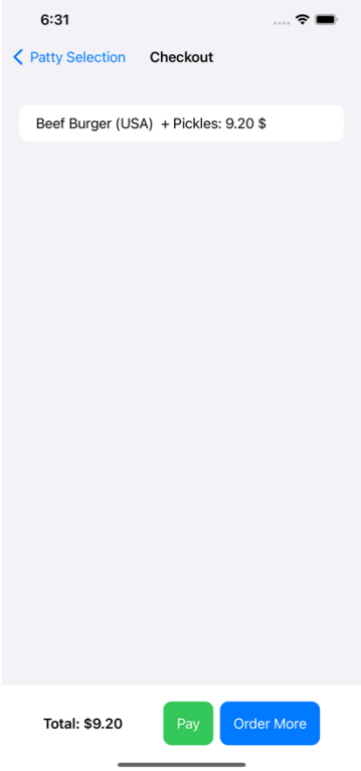


Abbildung 7 Wenn man bei Select-Type oder im Zutatenmenü auf den Cart-Button klickt, landet man im Checkout

Nachdem klicken auf “Pay” landet man wieder auf dem Startbildschirm, bei “Order More” landet man auf dem Patty-Select Bildschirm. Der Test ist insoweit nicht bestanden, dass nachdem Zutatenmenü das Checkout nicht angezeigt wird und man auf den vorherigen Menüs auf den Cart-Button klicken muss, damit man zum Checkout gelangt.

Der Test ist nicht bestanden.

Test 2 Burger wird richtig erstellt

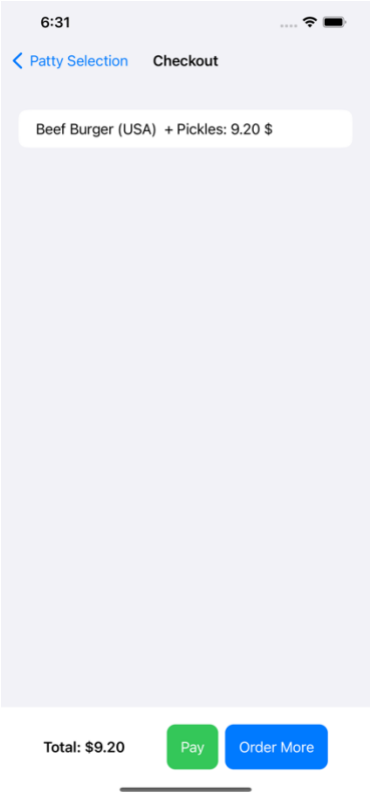


Abbildung 8 Checkout Ansicht

Dieser Test gilt als bestanden, wenn der kreierte Burger der Bestellung entspricht und der Preis korrekt ist. Dies ist der Burger von Test 1 und der wird korrekt angezeigt und der Preis stimmt auch. Für den Preis wurde $5 verlangt wegen dem Beefpatty, sowie $0.7 für die Essiggurken. Da der Benutzer als Land die USA ausgewählt hat, wird der Patty-Preis mit einem Faktor 1.7 multipliziert und man kriegt einen Preis von 5 * 1.7 + 0.7 = 9.2.

Der Test gilt als bestanden.

Test 3 Cart-Button Anzahl ist korrekt

Dieser Test gilt als bestanden, wenn die Anzahl an Burger in einer Bestellung der Zahl unten im Cart-Button entsprechen. In Test 1 konnte man schon sehen, dass der Cart-Button nach der Erstellung von einem Burger aktualisiert wurde. Dies wird auch immer richtig klappen, da der

Shop eine Liste mit allen bestellten Burgern führt. Die Liste ist dann verbunden mit dem Cart-Button und wird laufend aktualisiert, da es die Länge der Liste liest.

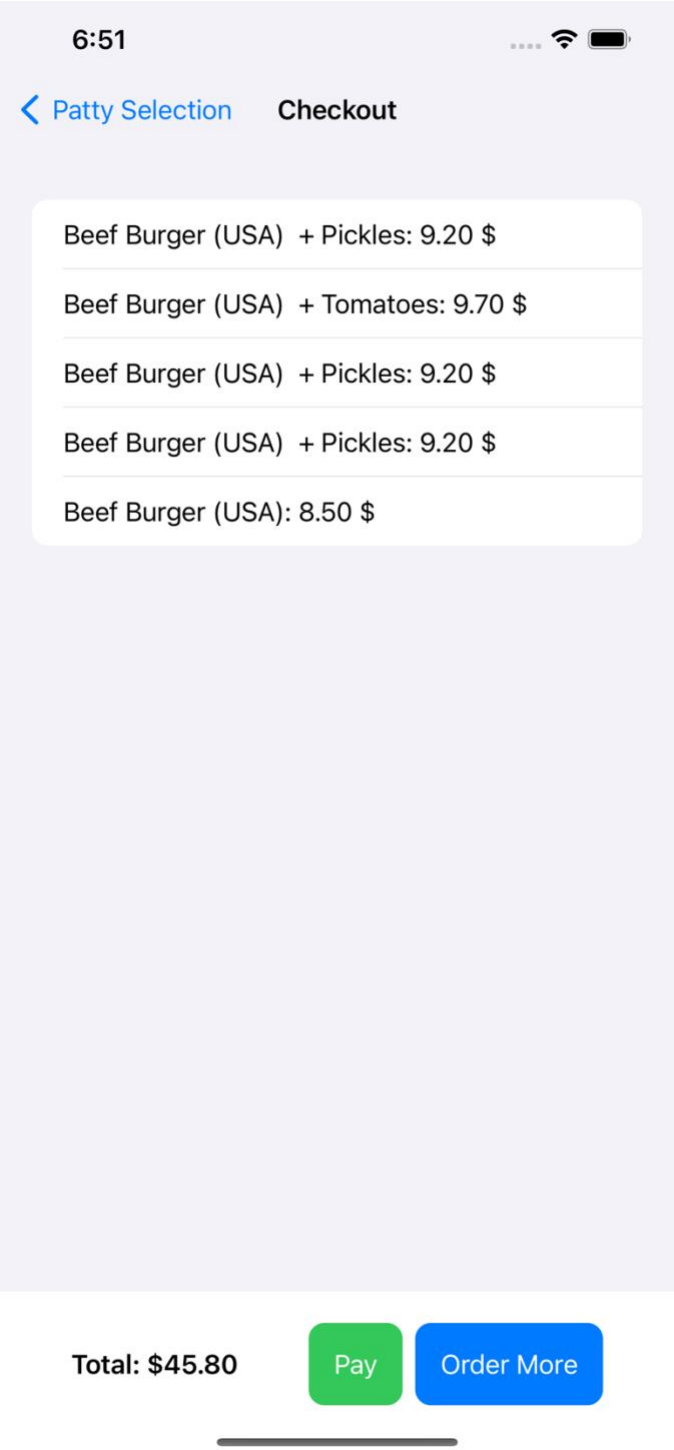


Abbildung 9 Checkout mit 5 Burgern

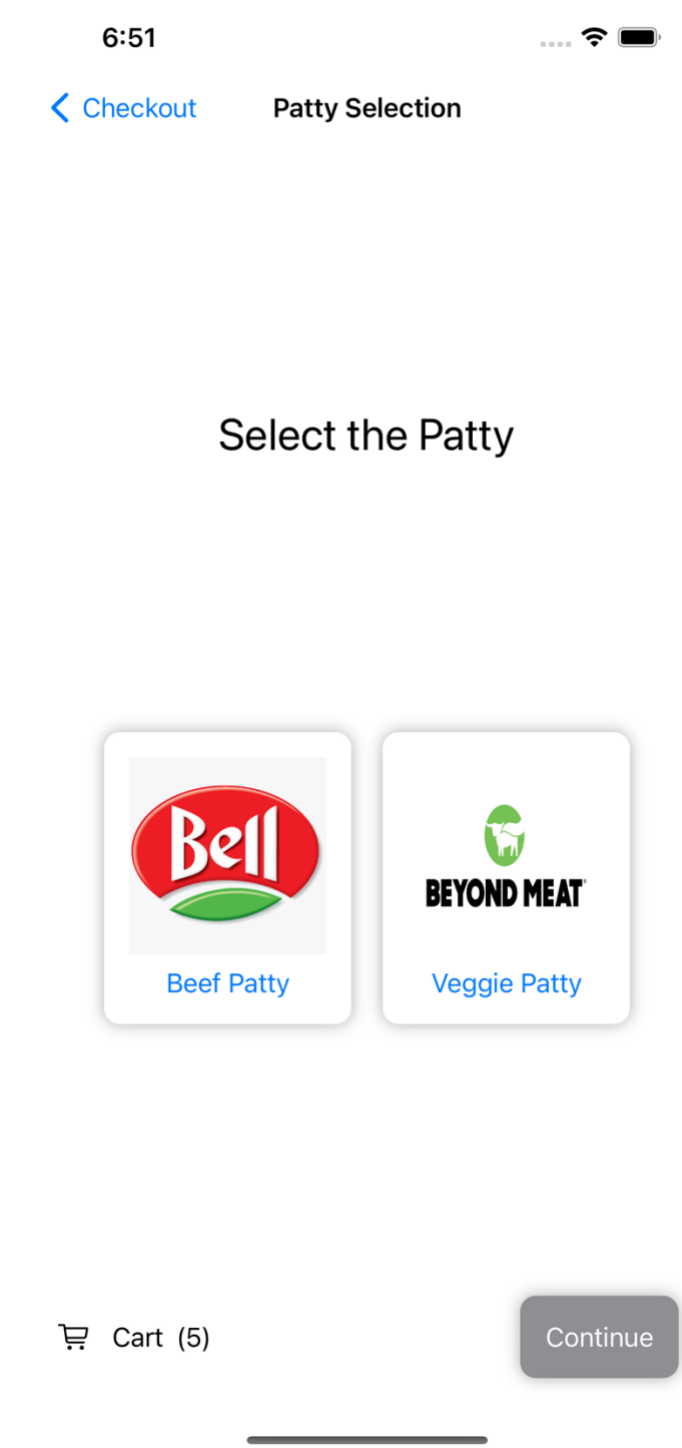
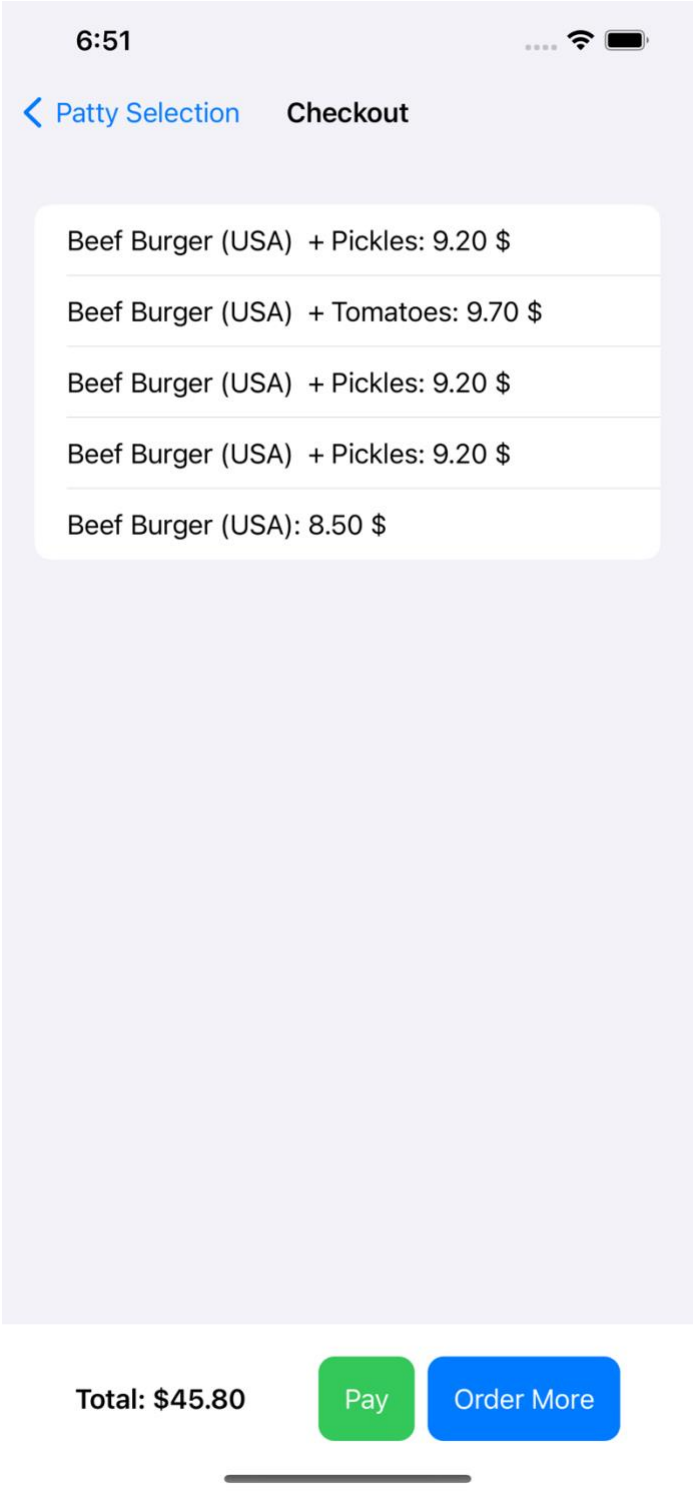


Abbildung 10 Cart hat 5 Items

8..

Test 4 Checkout Menu wird mit allen Burgern angezeigt



Dieser Test gilt als bestanden, wenn alle bestellten Burger im Checkout drin sind.

Da dies in Abbildung 11 der Fall ist, gilt der Test als bestanden.

Abbildung 11 Checkout mit 5 Burgern

Test 5 Zurückgehen zwischen Menüs

Dieser Test gilt als bestanden, wenn man vom Zutatenmenü zurückgehen kann zum Patty-Select Bildschirm. Da dies wie in Abbildungen 12 und 13 sichtbar ist, klappt. Gilt der Test als bestanden. Die Bestellungen bleiben auch korrekt, da noch keine Daten abgeschickt werden falls man einen Patty-Typ auswählt sondern nur eine Variable überschrieben wird. Ein Burger wird auch kreiert, nachdem man beim Zutatenmenü auf «Order» drückt.

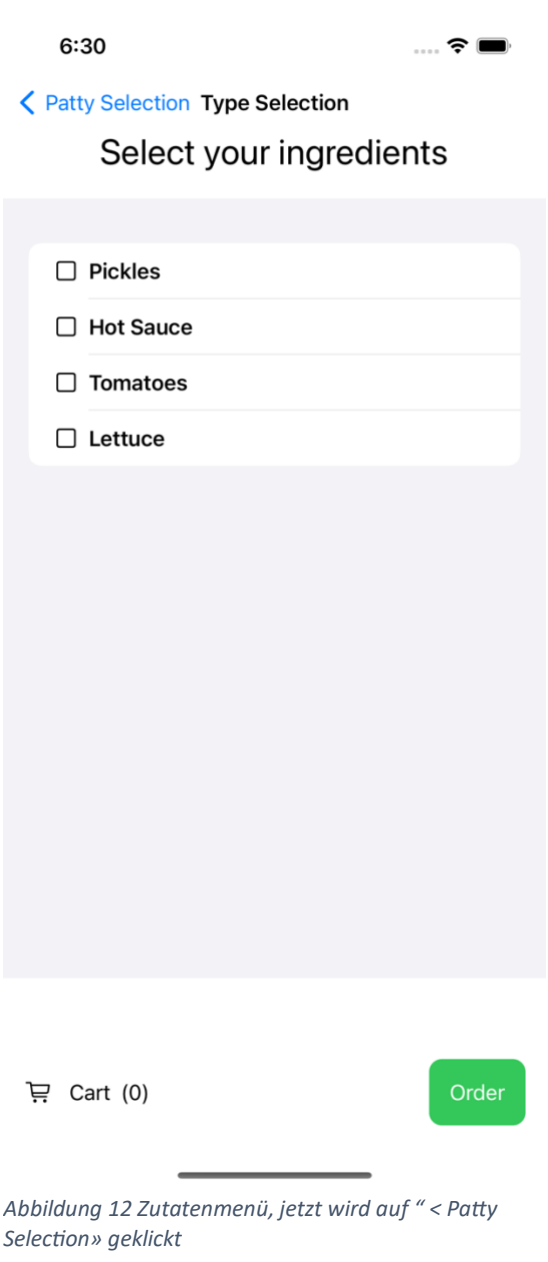


Abbildung 12 Zutatenmenü, jetzt wird auf " < Patty Selection» geklickt

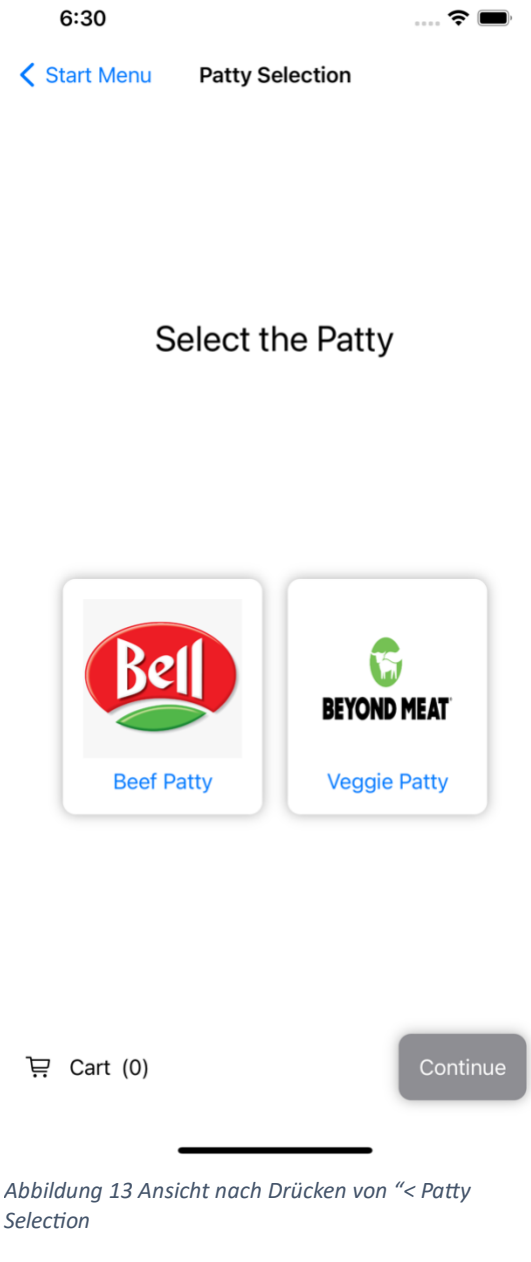


Abbildung 13 Ansicht nach Drücken von "< Patty Selection

User Experience

Die User Experience ist sehr mässig, da die Bilder nicht grossartig aussagekräftig sind und es viel verschwendeten Platz hat.

Wenn ein Patty ausgewählt wird im Menü, wird es nur kurz grau und wird nicht weiter angezeigt. Das heisst zum sicherstellen, dass man das richtige hat, müsste man es nochmals anklicken. Das Checkout Menü erinnert auch einen zu sehr an die Einstellungsapp in iOS und hat wenig schöne Details. Es ist ausserdem unübersichtlich und Icons könnten die Sicht vereinfachen.

Positiv ist jedoch, da wir mit native Technologien arbeiten, dass der Text sich den Benutzereinstellungen anpasst und dass das Menü auch von VoiceOver erkannt werden kann. Dies ist ein sehr grosser Vorteil von SwiftUI, es ist sehr gut integriert mit den IPhone-Technologien und Accesibility-Einstellungen.

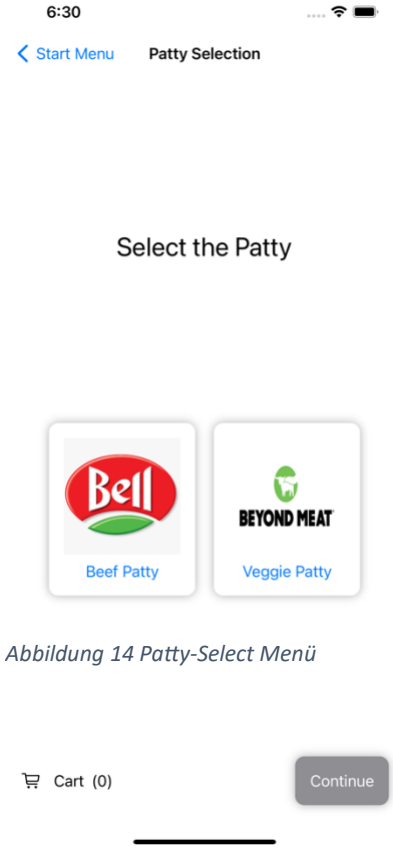


Abbildung 14 Patty-Select Menü

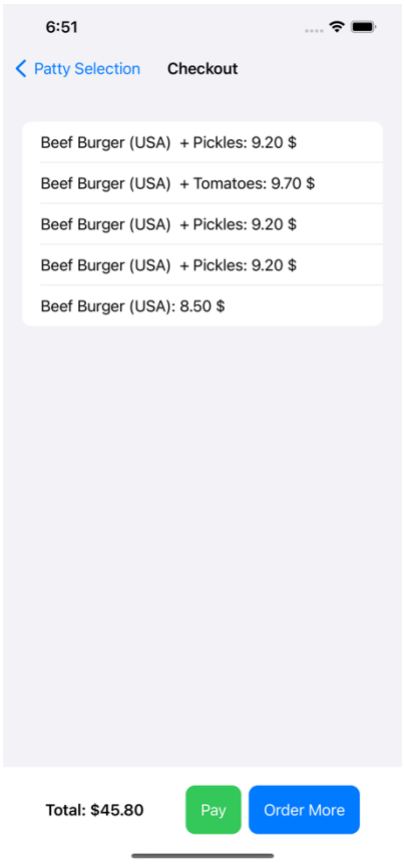


Abbildung 15 Checkout Ansicht

Fazit

Als Fazit kann ich sagen das Swift und SwiftUI wundervolle Technologien sind, jedoch sind sie immens komplex und haben eine Steile Lernkurve. Das hat mir die Arbeit sehr erschwert und

dafür gesorgt, dass ich sehr viel Zeit mit einlesen, recherchieren und Troubleshooting verbracht habe. Falls man aber nur für IPhones entwickeln will, scheint SwiftUI jedoch die beste Option zu sein, gegeben, XCode funktioniert und man hat die nötige Zeit um sich einzuarbeiten.

Alles in allem bin ich zufrieden mit der Arbeit, hoffe jedoch eine komplexere App bauen zu können falls ich wieder etwas ähnliches mache.